MULTIDIMENSIONAL DATA MODEL FOR DATA WAREHOUSES

G. Velinov[1], M. Kon-Popovska[2]

[1]IT Department, ESM, Orce Nikolov bb, Skopje

[2]Institute of Informatics, Faculty of Natural Science and Mathematics

Sts. Cyril and Methodius University

Arhimedova bb, P.O.Box 162, Skopje, Macedonia

goranv@ii.edu.mk; margita@ii.edu.mk

Abstract: Data Warehouses (DW) and Online Analytical Processing (OLAP) are essential elements of Decision Support Systems (DSS), they enable business decision makers to creatively approach, analyze and understand business problems.

OLAP data is frequently organized in the form of multidimensional data cubes each of which is used to examine a set of data values, called facts. Each fact is combination of multiple dimensions with multiple levels per dimension. The goal of this paper is the introduction of a multidimensional data model. The model is able to represent and capture natural hierarchical relationships among members (attributes) within a dimension. Moreover the data model is able to represent the relationships between dimension members and facts by mean of cube cells.

Keywords: data warehouses, OLAP, multidimensional data model, data cube

1. Introduction

In this paper we present mathematical theory of logical organization of Multidimensional data model. We discuss two similar approaches.

The first approach described in [3], has strong relation with mathematics by applying a set of new defined mathematic concepts, i.e. *H-relation*, *H-element*, *H-set*, *H-graph* and *H-partition*. The mathematic soundness provides a foundation to handle natural hierarchical relationships among data elements within dimensions with many levels of complexity in their structures. Afterwards, the multidimensional data model organizes data in the form of metacubes. Instead of containing a set of data cells, each metacube is associated with a set of groups each of which contains a subset of the data cell set. Furthermore, metacube operators (e.g. *jumping*, *rollingUp* and *drillingDown*) are defined in a very elegant manner. This approach is presented in Section 2, named "Conceptual Data Model".

The second approach described in [5] has incorporated dimensions as major entities in query and update languages. In the logical layer, a dimension is composed of a schema and an instance. The dimension schema includes a directed acyclic graph (DAG) of levels, called hierarchy schema, where levels may have attributes associated with them. Levels can be viewed as foreign keys of tables containing values for the attributes. On the other hand, a dimension instance consists of a set of members for each level, called member sets; and a hierarchy relation that models the ancestor/descendant relation between the members. A dimension schema is homogeneous if for every pair of levels l_1 and l_2 such that l_1 rolls up to l_2 we have that in every dimension instance conveyed in the schema, the rollup function is a total function from the member set of l_1 to the member set of l_2. A dimension schema is strictly homogeneous if it is homogeneous and it has only one level that is at the bottom of the hierarchy schema; a dimension schema is heterogeneous if it is not homogeneous. This approach is formally described in Section 3, named "Multidimensional Data Model".

2. Conceptual Data Model

In this approach, a multidimensional data model is constructed based on a set of dimensions $D=\{D_1,\ldots,D_x\}$, $x \in N$, a set of measures (facts) $M=\{M1,\ldots,M_y\}$, $y \in N$ and a set of metacubes $C=\{C_1,\ldots,C_z\}$, $z \in N$, each of which is associated with a set of groups $Groups(C_i)=\{G_1,\ldots,Gp\}$, $p,i \in N$, $1 \le i \le z$. The following sections formally introduce the descriptions of dimensions with their structures, measures, metacubes and their associated groups.

2.1 The Concepts of H-Set

Definition 2.1.1. [*H-relation*] A binary relation $\prec^*$ on a set S is called *a H-relation* if $\prec^*$ is:

- *irreflexive:* if, $(x,x) \notin R$, for all $x \in S$.
- *transitive:* if $(x,z) \in R$, and $(z,y) \in R$, then $(x,y) \in R$, for all $x,y,z \in R$.

Definition 2.1.2. [*H-set and H-element*] A *H-set* is a pair (S, $\prec^*$) and satisfies the following conditions:

Each element of S is called *H-element*, therefore the set S is now denoted by $S=\{he_i,\ldots,he_j\}$.

Having only one *root* element: $\exists! root \in S : (\neg \exists he_p \in S : he_p \prec * root)$,

Having leaf elements: $\{(he_t \in S) \mid (\neg \exists he_u \in S : he_t \prec * he_u)\}$,

An *H*-graph $G_H=(V_H, E_H)$ representing the H-set (S, $\prec*$) is a directed graph that is defined as follows:

$-V_H=S$. A vertex represents each element of the set S. There exists only one *root* vertex, which is the vertex for the *root* element.

$-EH \subseteq S \times S$. If $he_i \prec * he_j$ for distinct elements he_i and he_j, then the vertex for he_i is positioned higher than the vertex for he_j; and if there is no he_k different from both he_i and he_j such that $he_i \prec * he_k$ and $he_k \prec * he_j$, then an edge is drawn from the vertex he_i downward to the vertex he_j.

Definition 2.1.3. [*H-path*] Let G_H be a *H-graph* representing a *H-set* (S, $\prec*$). A *H-path hp* in G_H is a nonempty sequence $hp=(he_0,...,he_l)$ of vertexes such that $(he_i, he_{i+1}) \in E_H$ for each $i=[0,l-1]$.The H-path *hp* is from he_0 to he_l and has length l.

Definition 2.1.4. [*H-partition*] A collection of distinct non-empty subsets of a set S, denoted by $\{S_1,...,S_l\}$, $l \in \mathrm{N}$ is a *H-partition* of S if the three following conditions are satisfied:

- $S_i \cap S_j = \Theta, \forall i, j \in \mathrm{N}, 1 \le i \le j \le l$,
- $\bigcup_{i=1}^{l} S_i = S$,
- $\forall S_i, 1 \le i \le l : (\neg \exists he_t, he_u \in S_i) \mid ((he_t \prec * he_u) or (he_u \prec * he_t))$.

2.2 The Concepts of Dimensions

First are introduced hierarchical relationships among dimension members by means of one hierarchical domain per dimension. A hierarchical domain is a *H-set* of dimension elements, organized in hierarchy of levels, corresponding to different levels of granularity. It also allows us to consider a dimension schema as a *H-set* of levels. In this concept, a dimension hierarchy is a *H-path* along the dimension schema, beginning at the root level and ending at a leaf level. Moreover, the definitions of two dimension operators, namely $O_E^{ancestor}$ and $O_E^{descendant}$, provide abilities to navigate along a dimension structure. In a consequence, dimensions with any complexity in their structures can be captured with this data model.

Definition 2.2.1. [*Dimension Hierarchical Domain*] A dimension hierarchical domain, denoted by *dom*(D)=<*Delements*(D), $\prec_E^*$), is a *H-set*, where:

DElements(D)=$\{all\} \cup \{dm_1,...,dm_n\}$ is a set of dimension elements of the dimension D.

The *all* is the *root*,

The binary relation $\prec_E^*$ on the set *DElements*(D) is a *H-relation*.

Definition 2.2.2. [*Dimension Schema*] A dimension schema is a *H-set* of levels,

denoted by *DSchema*(D)= $\langle Levels(\mathrm{D}), \prec_L^* \rangle$, where:

- *Levels*(D)=$\{all\} \cup \{l_1,...,l_h\}$, $h \in \mathrm{N}$ is a finite set of levels of a dimension D, where:

$-\forall l_i \in Levels(\mathrm{D})$), l_i=<$Lname_i$, $dom(l_i)$>: $Lname_i$ is the name of a level,

$Dom(l_i)$ is one of $\{dom(all), dom(l_i),...,dom(l_h)\}$, $h \in \mathrm{N}$, the collection of which is a *H-partition* of *DElements*(D),

The *All* is the *root level*, where: $dom(All)=\{all\}$,

– Leaf levels: $\{\forall l_i \in Levels(\mathrm{D}) \,|\, (\forall dm_j \in dom(l_i)$ is a *leaf element*$)\}$.

The binary relation $\prec_L^*$ on the set *Levels*(D) is a *H-relation* and satisfies the following condition:

$\forall l_i, l_j \in Levels(\mathrm{D}), l_i \prec_L^*$ is given if there exits a map $f_{ances} : dom(l_j \to dom(l_j)$:

$(\forall dm_p \in dom(l_j)), (\exists! dm_q \in dom(l_i) \,|\, dm_q = f_{ances}(dm_p))$, such: $dm_p \prec_E^* dm_q$.

Definition 2.2.3. [*Dimension Hierarchy*] Let G_H^L be a *H-graph* representing the *H-set DSchema*(D)= $\langle Levels(\mathrm{D}), \prec_L^* \rangle$, which is the schema of a dimension D. A hierarchy is a *H-path hp* =(*All*,...,l_{leaf}) that begins at the *All* (*root*) level and ends at a leaf level. Let *H*(D)=$\{h_1,...,j_m\}$ $m \in \mathrm{N}$ be a set of hierarchies of a dimension D. If *m*=1 then the dimension has single hierarchical structure, else the dimension has multihierarchical structure.

Definition 2.2.4. [*Dimension Operators*] Two dimension operators (*DO*), namely $O_E^{ancestor}$ and $O_E^{descendant}$, are defined as follows: $\forall l_c, l_a, l_d \in Levels(\mathrm{D}), \forall dm_i \in dom(l_c)$:

$$O_E^{ancestor}(dm_i, l_a) = \begin{cases} dm_j \in dom(l_a) : dm_j \prec_E^* dm_i & \text{If } (l_a \prec_L^* l_c) \\ undefined & \text{Else} \end{cases}$$

$$O_E^{descendant}(dm_i, l_d) = \begin{cases} dm_t \in dom(l_d) : dm_i \prec_E^* dm_t & \text{If } (l_c \prec_L^* l_d) \\ undefined & \text{Else} \end{cases}$$

2.3 The Concepts of Measures

In this section are discussed measures, which are the objects of analysis in the context of multidimensional data model. First, is defined measure schema, which is a tuple $Mschema(\mathrm{M}) = \langle Fname, \mathrm{O} \rangle$. In case a measure that O is ”NONE”, then the measure stands for a fact, otherwise it stands for an aggregation.

Definition 2.3.1. [*Measure Schema*] A schema of a measure M is a tuple $Mschema(\mathrm{M}) = \langle Fname, \mathrm{O} \rangle$, where:

- *Fname* is a name of a corresponding fact,
- $\mathrm{O} \in \Omega \cup \{\mathrm{NONE}, \mathrm{COMPOSITE}\}$ is an operation type applied to a specific fact. Furthermore:

– $\Omega = \{\mathrm{SUM}, \mathrm{COUNT}, \mathrm{MAX}, \mathrm{MIN}\}$ is a set of aggregation functions.

– COMPOSITE is an operation (e.g. average), where measures cannot be utilized in order to automatically derive higher aggregations.

– NONE measures are not aggregated. In this case, the measure is the fact.

Definition 2.3.2. [*Measure Domain*] Let $\mathcal{N}$ be a numerical domain where a measure value is defined (e.g. **N**, **Z**, **R** or a union of these domains). The domain of a measure is a subset of $\mathcal{N}$. We denote by .) M ($dom(\mathrm{M}) \subset \mathcal{N}$

2.4 The Concepts of MetaCubes

First, a metacube schema is defined by a triple of a metacube name, an *x* tuple of dimension schemas, and a *y* tuple of measure schemas. Afterwards, each data cell is an intersection among a set of dimension members and measure data values, each of which belongs to one dimension or one measure. Furthermore, data cells of within a metacube domain are grouped into a set of associated granular groups, each of which expresses a mapping from the domains of *x*-tuple of dimension levels (independent variables) to *y*-numerical domains of *y*-tuple of numeric measures (dependent variables). Hereafter, a metacube is constructed based on a set of dimensions, and consists a metacube schema, and is associated with a set of groups.

Let a metacube C be constituted from *x* dimensions D1,..,D_x, $x \in \mathrm{N}$ and *y* measures M1,.., My, $y \in \mathrm{N}$.

Definition 2.4.1. [*MetaCube Schema*] A metacube schema is tuple *CSchema*(C)=$\langle$ *Cname, Schemas, MSchemas* $\rangle$:

- *Cname* is the name of a metacube,

DSchemas =<*Dschema*(D1),…, *Dschema*(D1)> is a x-tuple of schemas of x dimensions D1,.., Dx with *Dschemas*(i)=*Dschema*(D$_i$), $1 \le i \le x$,

DSchemas =<*Mschema*(M1),…, *Mschema*(M$_y$)> is a y-tuple of schemas of y measures M1 ,.., My, $y \in$ N with *Mschemas*(j)=*Dschema*(M$_j$), $1 \le j \le y$.

Definition 2.4.2. [*MetaCube Hierarchy Domain*] A metacube hierarchy domain, denoted by *Dom*(C)= $\langle$ *Cells*(C), $\prec_C^*$ $\rangle$ is a H-set, where:

Given a function $f : \mathop{\mathrm{X}}_{i=1}^{x} dom(\mathrm{D}_i) \mathrm{x} \mathop{\mathrm{X}}_{\mathrm{j}=1}^{\mathrm{y}} dom(\mathrm{M}_j) \rightarrow \{false,\ true\}$, *Cells*(C) is determined as: $Cells(\mathrm{C}) = \{c \in \mathop{\mathrm{X}}_{i=1}^{x} dom(\mathrm{D}_i) \mathrm{x} \mathop{\mathrm{X}}_{\mathrm{j}=1}^{\mathrm{y}} dom(\mathrm{M}_j) \mid f(c) = true\}$

The binary relation $\prec_C^*$ on the set *Cells*(C) is a *H-relation*.

Definition 2.4.3. [*Group*] A group is triple G=$\langle$ *Gname*, *Gschema*(G) *dom*(G) $\rangle$ where:

- Gname is the name of the group,
- GSchema(G)=$\langle$ Glevels(G), GMschema(G) $\rangle$:

$GLevels(\mathrm{G}) =< l_{D_1},...,l_{D_x} > \in \mathop{\mathrm{X}}_{i=1}^{x} Levels(\mathrm{D}_i)$ is a x-tuple of levels of the x dimensions D1,.., D$_x$, $x \in$ N .

GMSchemas(G) =<*MSchema*(M1),…, *MSchema*(M$_y$)> y-tuple of measure schemas of the y measures M1,..,M$_y$, $y \in$ N.

$$d\mathrm{om}(\mathrm{G}) = \{c \in \mathop{\mathrm{X}}_{i=1}^{x} dom(l_{\mathrm{D_i}}) \mathrm{x} \mathop{\mathrm{X}}_{\mathrm{j}=1}^{\mathrm{y}} dom(\mathrm{M}_j) \in Cells(\mathrm{C})\}$$

Let h_i be a number of levels of each dimension D$_i$ $1 \le i \le x$.The total set of groups over a metacube C is defined as *Groups*(C) ={G$_1$,…,G$_p$}, $p = \prod_{i=1}^{x} h_i$.

Definition 2.4.4. [*MetaCube Operators*] Three basic navigational metacube operators (*CO*), namely *jumping*, *rollingUp* and *drillingDown*, which are applied to navigate along a metacube C, orresponding to a dimension D$_i$, are defined as follows:

$\forall \mathrm{G}_c \in Groups(\mathrm{C}), l_c \in Levels(\mathrm{D}_i)$ and $l_c \in GLevels(\mathrm{G}_c), \forall l_j, l_r, l_d \in Levels(\mathrm{D}_i)$.

jumping:

$\text{jumping}(G_c,l_j,D_i) = G_j = \langle \text{GLevels}(G_j), \text{GMSchemas}(G_j)\rangle$

Where:

$$GMSchemas(G_j) = GMSchemas(G_c)$$

$$GLevels(G_j)(i) = l_j,\ GLevels(G_j)(k) = GLevels(G_c)(k),\ \forall k \neq i\,.$$

rollingUp:

$\forall dm \in dom(l_c),\ G_r = jumping(G_c, l_r, D_i).$

$$rollingUp(G_c, dm, l_r, D_i) = < GSchema(G_r^{sub}), dom(G_r^{sub}) >$$

Where:

$$GSchema(G_r^{sub}) = GSchema(G_r),$$

$$dom(G_r^{sub}) = \{c_r \in dom(G_r) \mid \exists c \in dom(G_c) : c.dms(i) = dm,$$

$$c_r.dms(i) = O_d^{ancestor}(dm, l_r, G_i), c_r.dms(j) = c.dms(j), \forall j \neq i\}$$

drillingDown:

$\forall dm \in dom(l_c),\ G_r = jumping(G_c, l_d, D_i).$

$$DrillingDown(G_c, dm, l_d, D_i) = < GSchema(G_d^{sub}), dom(G_d^{sub}) >$$

Where:

$$GSchema(G_r^{sub}) = GSchema(G_r),$$

$$dom(G_d^{sub}) = \{c_d \in dom(G_d) \mid \exists c \in dom(G_c) : c.dms(i) = dm,$$

$$c_d.dms(i) = O_d^{ancestor}(dm, l_d, D_i), c_d.dms(j) = c.dms(j), \forall j \neq i\}$$

Definition 2.4.5. [*Metacube*] A metacube is a tuple C=< *Cschema*, $\mathcal{D}$, *Groups*, *CO* > where:

- *CSchema* is a metacube schema,
- $\mathcal{D}$= D1,..,D_x, $x \in$ N is the set of dimensions,
- *Groups* is a total set of groups of the metacube.

CO is a set of metacube operators.

3. Multidimensional Data Model

In this section, we present second framework for modelling dimensions. A dimension schema will consist of a hierarchy schema and a set of constraints.

3.1 Hierarchy Schema

A hierarchy schema allowed heterogeneity, multiple hierarchical paths, and multiple bottom levels. Multiple hierarchical paths are frequently required. Allowing multiple bottom levels makes it possible to have more natural schemas in several situations.

Consider a set of members **E**, a set of levels **L**, and set of attributes **A**.

Definition 3.1.1. (Hierarchy Schema). A hierarchy schema is a tuple $G=(L, \nearrow, A, \sigma)$, where $L \in \mathbf{L}$ is a set of levels with a distinguished level All; $\nearrow$ is a binary relation on L such that $(L; \nearrow)$ conforms a rooted DAG with *all* as root (we denote by $\nearrow^*$ the transitive closure of $\nearrow$); $A \subseteq \mathbf{A}$ is a set of attributes; and $\sigma : L \rightarrow 2^A$ assigns to each level a set of attributes.

We refer to the set of bottom levels of a hierarchy schema, $\{l \in L \mid \neg\exists l' \in L : l' \nearrow l\}$, as L_{Bottom}. Given two levels $l_a, l_b \in L$, we denote by Υ_{l_a, l_b} the set of paths between l_a and l_b in G.

3.2 Dimension Instance and Schema

An instance of a dimension is obtained by specifying a set of members for each level, along with the descendant/ancestor relation < between them.

Definition 3.2.1. (Dimension Instance). A dimension instance is a tuple $(G, E, <, T)$, where G is a hierarchy schema; E is a set of sets of members in **E**, one set E_l for each level $l \in L$, where we denote by E_d the union of the member sets of a dimension d; < is a relation between members conforming a rooted DAG with root *all*, where we denote by << the transitive closure of <; and finally, T is a set of relations that contains one relation T_l, with attributes $\sigma(l) \cup \{l\}$, for every level $l \in L$. The following conditions hold: (1) the member set of a level l, E_l, is the active domain of l in T_l, and l is a key of T_l; (2) for every pair of members e_a, e_b such that $e_a \in E_{l_a}$, $e_b \in E_{l_b}$ and $e_a < e_b$, we have $l_a \nearrow l_b$, and there are no members $e_1, \ldots, e_n \in E_d$ such that $e_a < e_1 < \ldots e_n < e_b$; (3) for every member $e \in E_d$ such that e $e \neq all$ we have $e << all$.

The first condition says that the relation for level l contains exactly one tuple for each element in E_l. The second condition essentially says that the edges in the dimension hierarchy represent links between levels, that must exist whenever we have a direct descendant/ancestor relation between some pair of members in the levels. And finally, the last condition states that all the members reach the top member *all*. Note that the member sets are not necessarily disjoint.

Given a dimension *d*, a *leaf member* is a member $e \in E_d$ with no descendant members. An important feature of the model is that it may have leaf members in non-bottom levels. This allows updating the dimension (for instance, we might want to add a city but do not yet have stores that belong to it.) However, in order to simplify the presentation, we make two assumptions: (a) all the leaf members belong to the bottom levels; and (b) the member sets of the bottom levels are pairwise disjoint. We define a *base level*, denoted by l_{base}, containing the union of the member sets of the bottom levels. The results in this paper can be extended to dimensions where (a) and (b) does not hold by a more detailed treatement of base members, which basically consists in defining them as id's of the leaves.

Definition 3.2.2. (Rollup Operators). Given a dimension instance *h*, we define the direct rollup operator, that takes a dimension and two of its levels l_1, l_2 and gives a relation with attributes l_1 and l_2 defined as follows: $d\Gamma_{l_1}^{l_2} = \{(x_1, x_2) \mid x_1 \in E_{l_1} \wedge x_2 \in E_{l_2} \wedge x_1 < x_2\}$. We have the rollup operator with the same signature of $d\Gamma$ which gives a relation with attributes l_1 and l_2 defined as follows: $\Gamma_{l_1}^{l_2} = \{(x_1, x_2) \mid x_1 \in E_{l_1} \wedge x_2 \in E_{l_2} \wedge x_1 << x_2\}$. The base rollup operator takes a level *l* a defines the relation, with attributes l_{base} and *l*, that groups the base elements to it, and is defined as follows: $\Gamma_{l_{base}}^{l} = \{(x, y) \mid x \in E_{l_{base}} \wedge y \in E_l \wedge x << y\}$.

The following are some basic properties of the rollup operators: given a dimension instance *d* we have: (a) if $\neg(l_a \nearrow l_b)$ then $d\Gamma_{l_a}^{l_b} = \Theta$; (b) if $\neg(l_a \nearrow^* l_b)$ then $\Gamma_{l_a}^{l_b} = \Theta$; (c) if $\Upsilon_{l_a, l_b} = \{l_a l_b\}$ then $\Gamma_{l_a}^{l_b} = d\Gamma_{l_a}^{l_b}$.

Definition 3.2.3. (Partitioned Instances). A dimension instance is *partitioned* when all its rollup relations are single valued (partial functions). The partitioning property appears as an inherent constraint in the dimension models. It requires that each member in the base level reach, through $\Gamma_{l_{base}}^{l}$, not more that each level *l*. In this sense, each level represents partitioned classification of the base members. In the sequel we assume that all dimension instances are partitioned.

We are ready to define dimension schema as a hierarchy schema plus a set of constraints in some constraint language *CL*.

Definition 3.2.4. (Dimension Schema). A *CL-dimension schema* is a tuple ds =(*G*, $\sum$) where *G* is a hierarchy schema; and $\Sigma \subseteq CL$

The constraint language must have a notion of satisfaction, denoted by $|=_{CL}$. A dimension instance *d* is over a dimension schema *ds* if $G_d = G_{ds}$, $d \models_{CL} \sum$.

Given a dimension schema *ds* we denote by *I*(*ds*) the set of dimension instances that are over *ds*.

3.3 Classes of Dimension Schemas

Definition 3.3.1. (Homogeneous and Heterogeneous Dimension Schemas). A dimension schema ds is *homogeneous* if every dimension *d* over *ds* satisfies: for every pair of levels l_1, l_2 such that $l_1 \nearrow l_2$ we have that $\Gamma_{l_1}^{l_2} : E_{l_1} \rightarrow E_{l_2}$ is a total function. A dimension schema is *strictly homogenous* if it is homogeneous, and has a single bottom level. A dimension schema is *heterogeneous* if it is not homogeneous.

In a homogeneous dimension instance, the ordering of the levels provided by the graph is exactly the same as the ordering of the grains represented by the levels. In other words, as we move up the hierarchy schema we reach levels that represent coarser partitions of the base members. Furthermore, the rollup functions capture precisely the containment relation between the partitions of levels connected in the graph. These two properties can no longer be true in heterogeneous dimension instances.

In order to formalize this intuition, let us define the notion of overlap, as introduced for classification schemas in statistical databases. The overlap between two levels l_1 and l_2 is a relation $\Theta_{l_1}^{l_2}$ with signature $E_{l_1} \mathbf{x}\, E_{l_2}$ that has an edge between the members e_1 and e_2 iff the intersection of the sets of base members that reach them is non-empty. The overlap between two levels can be defined with the following relational-algebra expression: . $\Theta_{l_1}^{l_2} = \pi_{l_1,l_2}(\Gamma_{l_{base}}^{l_1} \rhd\lhd \Gamma_{l_{base}}^{l_2})$. We say that $l_1 \leq l_2$ in the dimension instance d if $\Theta_{l_1}^{l_2} = \Gamma_{l_1}^{l_2}$. That is, $l_1 \leq l_2$ means: (a) the grain of l_1 is finer than the grain of l_2 because the overlap is a function (possibly partial); and (b) the rollup function $\Gamma_{l_1}^{l_2}$ captures precisely the containment relation between the grains of l_1 and l_2.

It is important to note, that in general $\leq$ is not transitive, i.e., if a dimension instance *h* satisfies $l_1 \leq l_2$, and $l_2 \leq l_3$ it does not necessarily satisfy $l_1 \leq l_3$.

Definition 3.3.2. (Hierarchical Dimension Schemas). A dimension schema ds is *hierarchical* if for every pair of levels $l_1, l_2 \in L_{ds}$ such that $l_1 \nearrow l_2$ we have that every dimension instance $d \in I(ds)$ satisfies $l_1 \leq l_2$. A dimension schema *ds* is strictly hierarchical if for every pair of levels $l_1, l_2 \in L_{ds}$ such that $l_1 \nearrow^* l_2$ we have that every dimension instance $d \in I(ds)$ satisfies $l_1 \leq l_2$.

The relationship among the classes of schemas is given by the following chain of inclusions: Strictly Homogeneous ⊂ Homogeneous ⊂ Strictly Hierarchical ⊂ Hierarchical .

3.4 Summarizability

In this section we extend the notion of summarizability to heterogeneous dimensions. A level represents a granularity of aggregation. In this sense, a level *l* can be associated with a one-dimensional cube view with no select conditions, that has the following form: $\Pi_{l,m=af(m)}(\Gamma^{l}_{l_{base}} d \triangleright\triangleleft F)$, where *d* is a dimension; *F* is a relation, called a fact table, with a special attribute *m* called the measure, and with l_{base} included in its attributes; af is an aggregate function; and *l* is a level. This cube view will be abbreviated as *cv*(*d*, *F*, *l*, *af*(*m*)).

Sumarizability of levels in a dimension is related to the correct derivation of one dimensional cube views from other one dimensional cube views using rollup relations for grouping. Such derivations involve queries of the form: $\Pi_{l,af^{c}(m)}(\bigcup_{i\in 1,\dots,n}(\pi_{l,m}\Gamma^{l}_{l_i} d \triangleright\triangleleft F_i))$. Intuitively, we are aggregating a set of fact tables $F_1,\dots,F_n$,using the rollup mappings $\Gamma^{l}_{l_1},\dots,\Gamma^{l}_{l_n}$.

Definition 3.4.1. (Summarizability). Given a dimension instance *d*, a set of levels $L=\{l_1,\dots,l_n\}$, and level *l, l* is summarizable from L in *d* if for every fact table *F*, and distributive aggregate function *af*, we have: $cv(d, F, l, af(m)) = \Pi_{l,af^{c}(m)}(\bigcup_{i\in 1,\dots,n}(\ \pi_{l,m}\Gamma^{l}_{l_i} d \triangleright\triangleleft cv(d, F, l, af(m))))$.Given a dimension schema *ds*, a set of levels $L=\{l_1,\dots,l_n\}$, and a level *l*, *l* Is summarizable from L in *ds* if *l* is summarizable from L in every instance *d* in *I*(*ds*).

The following lemma gives an alternative definition of summarizability.

Lemma 3.4.1. (Summarizability). A level l is summarizable from a set of levels $L=\{l_1,\dots,l_n\}$ in a dimension instance *d* iff $\Gamma^{l}_{l_{base}} = \bigcup_{i\in 1,\dots,n}\pi_{l_{base},l}(\Gamma^{l_i}_{l_{base}} \triangleright\triangleleft \Gamma^{l}_{l_i})$.

Recall thet we are assuming that the dimension is partitioned. From Lema 1 we have that a level *l* is summarizable from a single level l_1 in a dimension *d* iff $\Gamma^{l}_{l_{base}} = \Gamma^{l_1}_{l_{base}} \triangleright\triangleleft \Gamma^{l}_{l_1})$

4. Conclusion

In this paper we present two theoretical approaches for multidimensional data model, which facilitates even sophisticated constructs based on multidimensional data units or members such as dimension members, measure data values and then cells. Models are able to represent and capture natural hierarchical relationships among dimension members. Dimensions with complexity of their structures, such as: unbalanced and multihierarchical structures, can be modeled in an elegant and consistent way.

Moreover, the data models represent the relationships between dimension members and measure data values by mean of cube cells. In consequence, data cubes, which are basic components in multidimensional data analysis, and their operators are formally introduced.

One of the most important issues in data warehouse implementation is VIS (View and Index Selection) problem, as one aspect of choosing set of views and indexes to materialize. We will use the mathematical theory presented in these approaches as base to define search space of all possible views and indexes for VIS problem.

5. References

1. Poe, V. and Klauer, P. and Brobst S., "*Building a Data Warehouse for Decision Support",* Prantice Hall PTR, 1998
2. Bellatreche, L. and Karlapalem, K. And Mohania, M., "Some Issues in Design of Data Warehousing Systems", 1997
3. Binh, N.T. and Tjoa, A.M. and Wagner, R., "Conceptual multidimensional Data Model Based on MetaCube", 2000
4. Harinarayan, V. and Rajaraman, A. and Ullman, J.D., "Implementing Data Cubes Efficiently", Stanford University, 1996
5. Hurtado, C.A. and Mendelzon A.O., "Resoning About Summarazability in Heterogeneous Multidimensional Schemas", 2000